CEE 616: Probabilistic Machine Learning

M2 Linear Methods: L2b Logistic Regression

Jimi Oke

UMass Amherst

College of Engineering

Thu, Sep 25, 2025

Outline

Logistic regression

The logistic regression model has the form:

$$p(y|\boldsymbol{x};\boldsymbol{\theta}), \quad \boldsymbol{x} \in \mathbb{R}^D, y \in \{1, \ldots, C\} \tag{1}$$

- **Binary logistic regression:** $C = 2$:

$$p(y|\boldsymbol{x};\boldsymbol{\theta}) = \text{Ber}(y|\boldsymbol{\sigma}(\boldsymbol{w}^\top \boldsymbol{x})) \tag{2}$$

  where $\boldsymbol{\sigma}$ is the sigmoid function and $\boldsymbol{\theta} = \boldsymbol{w} = (b, w_1, w_2, \ldots, w_D)$

- **Multinomial logistic regression:** $C > 2$

$$p(y|\boldsymbol{x};\boldsymbol{\theta}) = \text{Cat}(y|\mathcal{S}(\boldsymbol{W}\boldsymbol{x})) \tag{3}$$

  where $\mathcal{S}$ is the softmax function and $\boldsymbol{\theta} = \boldsymbol{W}$.

Definitions

- Input vector: $\boldsymbol{x} = (1, x_1, x_2, \dots, x_D)$
- Weights (or weight vector): $\boldsymbol{w} = (b, w_1, w_2, \dots, w_D)$
- Bias: b (absorbed into weight vector)
- Sigmoid function:

$$\boldsymbol{\sigma}(\boldsymbol{w}^\top \boldsymbol{x}) = \frac{1}{1 + e^{-\boldsymbol{w}^\top \boldsymbol{x}}} \tag{4}$$

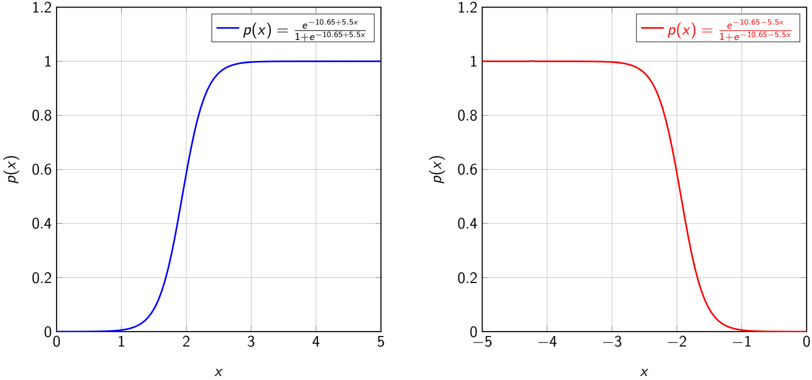
- Log-odds or logit: $\boldsymbol{w}^\top \boldsymbol{x}$ (binary case); $\boldsymbol{W}\boldsymbol{x}$ (multinomial)
- Softmax function:

$$\mathcal{S}(\boldsymbol{W}\boldsymbol{x}) = \left[\frac{e^{\boldsymbol{w}_1^\top \boldsymbol{x}}}{\sum_{c'=1}^{C} e^{\boldsymbol{w}_{c'}^\top \boldsymbol{x}}}, \cdots, \frac{e^{\boldsymbol{w}_C^\top \boldsymbol{x}}}{\sum_{c'=1}^{C} e^{\boldsymbol{w}_{c'}^\top \boldsymbol{x}}} \right] \tag{5}$$

where $\boldsymbol{W} = [\boldsymbol{w}_1, \boldsymbol{w}_2, \dots, \boldsymbol{w}_C]$ is a $C \times D$ weight matrix

Logistic (sigmoid) function

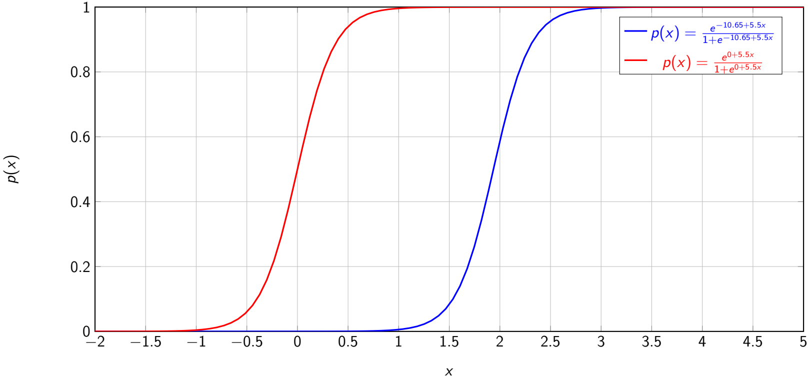
For $w_1 > 0$, the logistic function increases w.r.t. x. For $w_1 < 0$, the logistic function decreases w.r.t. x.



What happens when b is increased or decreased?

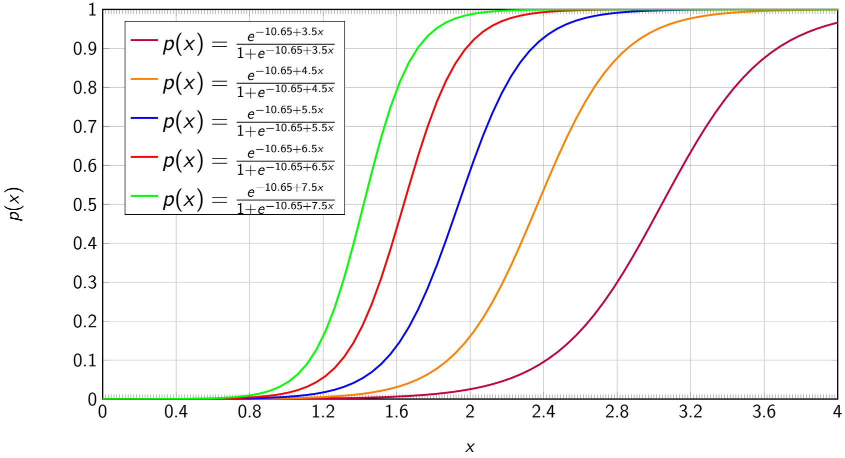
Logistic function (cont.)

b shifts the curve left or right (adjusts average fitted probabilities).



Logistic function (cont.)

w_1 adjusts the steepness of the curve: as β_1 increases, the curve becomes steeper



Odds ratio and the logit function

From the logistic function, we can obtain the **odds ratio** (OR) as:

$$OR = \frac{p(x)}{1 - p(\boldsymbol{x})} = e^{b + w_1 x} \tag{6}$$

which is considered as the relative likelihood of success ($p(x)$).

Taking the log of the odds ratio yields the log-odds or **logit** function:

$$logit(p(x)) = \log\left(\frac{p(\boldsymbol{x})}{1 - p(x)}\right) = b + w_1 x \tag{7}$$

Notes

- The logit function is linear in x
- The inverse of the logit function yields the logistic function
- In the generalized linear framework, logit is the *link function* between the predictors and the mean response

Logistic regression

Logistic regression is an approach for modeling the *probability* of a *multinomial* response.

In the simple case, we consider a binomial (or binary) response.

Logistic function

This is the model equation for simple logistic regression:

$$p(y = 1|\boldsymbol{x}; \boldsymbol{\theta}) = \frac{e^{b+w_1 x}}{1 + e^{b+w_1 X}} \tag{8}$$

where $p(y = 1|\boldsymbol{x}, \boldsymbol{\theta}) \in [0, 1]$

The logistic function is a member of the class of **sigmoid** functions (S-shaped curves) and can also be written:

$$p(y = 1|x, \boldsymbol{\theta}) = \frac{1}{1 + e^{-\boldsymbol{w}^\top \boldsymbol{x}}} = (1 + e^{-\boldsymbol{w}^\top \boldsymbol{x}})^{-1} = \sigma(\boldsymbol{w}^\top \boldsymbol{x}) \tag{9}$$

where $\boldsymbol{w}^\top = (b, w_1)$

Example 1: Binomial logistic regression with single predictor

Credit card defaults

- Data: A simulated data set containing information on ten thousand customers.
- Question: Can we predict which customers will default on their credit card debt (based on income, etc)?

Four variables:

- `default`: A factor with levels *No* and *Yes* indicating whether the customer defaulted on their debt
- `student`: A factor with levels *No* and *Yes* indicating whether the customer is a student
- `balance`: The average balance that the customer has remaining on their credit card after making their monthly payment
- `income`: Income of customer

Example 1: Binomial logistic regression (cont.)

We want to model the probability of `default` based on the `balance` predictor. The estimated coefficients from a computer program are:

```
              Estimate Std. Error z value Pr(>|z|)
(Intercept) -1.065e+01  3.612e-01  -29.49   <2e-16 ***
balance      5.499e-03  2.204e-04   24.95   <2e-16 ***
```

Note that the null hypothesis for the tests is: $H_0 : \boldsymbol{w}_i = 0$ (i.e. no dependence on the corresponding predictor)

Example 1: Binomial logistic regression (cont.)

The estimated model is:

$$\hat{p}(y=1|\boldsymbol{x};\boldsymbol{\theta}) = \frac{e^{(-10.65+0.0055x)}}{1+e^{(-10.65+0.0055x)}} = \frac{1}{1+e^{(10.65-0.0055x)}} = \sigma(1+e^{(10.65-0.0055x)}) \tag{10}$$

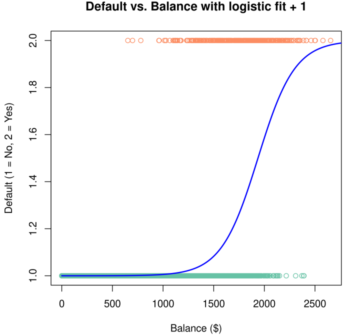


Figure: Logistic regression model predicting the probability of `default` based on

Example 1: Binomial logistic regression (cont.)

Recall the model:

$$\hat{p}(y = 1|x) = \frac{e^{-10.65+0.0055x}}{1 + e^{-10.65+0.0055x}}$$

1. How would $\hat{p}$ (the predicted probability) change if x were to increase by \$100?
2. What about if x were to decrease by \$100
3. There are 333 defaults out of 10000 observations. What is the impact of b?

Multiple logistic regression

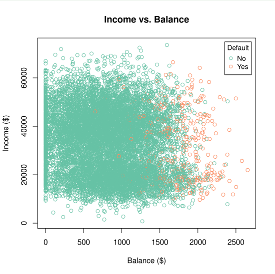
In multiple logistic regression, we predict a binary response using *multiple predictors*:

$$\log\left(\frac{p(x)}{1-p(x)}\right) = b + w_1 x_1 + \cdots + \boldsymbol{w}_D x_D = \boldsymbol{w}^\top X \tag{11}$$

where $\boldsymbol{x} = (x_1, \ldots, x_D)$.

Activity: Binomial logistic regression with multiple predictors

Using the `Default` dataset, predict the probability of `default` based on `balance` and `student`. Comment on your results and interpret the coefficient estimates.



Decision boundary

This is the line that defines the probability threshold τ for class assignment:
In 1-D:

$$x^* : p(y = 1 | x = x^*, \boldsymbol{\theta}) = \tau \tag{12}$$

Typically, $\tau = 0.5$

Estimation of logistic regression coefficients

The method of **maximum likelihood** is used to estimate logistic regression coefficients $\boldsymbol{w}$

- The likelihood function $\mathcal{L}(\theta)$ represents the support provided by a sample for a given parameter θ:

$$\mathcal{L}(\boldsymbol{\theta}) = \prod_{i=1}^{N} p_{c_i}(x_i; \boldsymbol{\theta}) \tag{13}$$

  - where $p_{c_i}(x_i; \boldsymbol{\theta}) = \Pr(G = c_i | X = x_i; \boldsymbol{\theta})$
  - In the two-class case: $\boldsymbol{\theta} = \boldsymbol{w} = \{b, w_1\}$

- Thus, we can write the conditional probabilities as:

$$p_1(x; \boldsymbol{\theta}) = p(x; \boldsymbol{\theta}) \tag{14}$$

$$p_2(x; \boldsymbol{\theta}) = 1 - p(x; \boldsymbol{\theta}) \tag{15}$$

- It is also convenient to encode c_i using a $0/1$ response y_i:

$$y_i = \begin{cases} 1, & \text{when } c_i = \text{Class 1} \\ 0, & \text{when } c_i = \text{Class 2} \end{cases} \tag{16}$$

Log-likelihood function for logistic regression

The principle of maximum likelihood dictates that the best parameter estimates are those that maximize the likelihood function.

- Equivalently, we *minimize* the **negative log-likelihood** function $\text{NLL}(\boldsymbol{\theta})$:

$$\text{NLL}(\boldsymbol{\theta}) = -\sum_i \log p_{c_i}(x_i; \boldsymbol{\theta}) \tag{17}$$

- In the binomial case, this simplifies to:

$$\text{NLL}(\boldsymbol{w}) = -\sum_i \left[y_i \log p(x_i; \boldsymbol{w}) + (1 - y_i) \log(1 - p(x_i; \boldsymbol{w}))\right] \tag{18}$$

- Recall that we model $p(x_i; \boldsymbol{w})$ as:

$$p(x_i) = \frac{e^{b + w_1 x_i}}{1 + e^{b + w_1 x_i}} \tag{19}$$

Log-likelihood function for logistic regression

Substituting (19) into (18), we obtain:

$$
\begin{aligned}
\mathrm{NLL}(\boldsymbol{w}) &= -\sum_i \left[y_i \log\left(\frac{e^{b+w_1x_i}}{1+e^{b+w_1x_i}}\right) + (1-y_i)\log\left(1-\frac{e^{b+w_1x_i}}{1+e^{b+w_1x_i}}\right)\right] \\
&= -\sum_i \left[y_i \log\left(\frac{e^{b+w_1x_i}}{1+e^{b+w_1x_i}}\right) + (1-y_i)\log\left(\frac{1}{1+e^{b+w_1x_i}}\right)\right] \\
&= -\sum_i \Big[y_i \log\left(e^{b+w_1x_i}\right) - y_i \log\left(1+e^{b+w_1x_i}\right) \\
&\qquad + \cancelto{0}{(1-y_i)\log(1)} - (1-y_i)\log\left(1+e^{b+w_1x_i}\right)\Big] \\
&= -\sum_i \Big[y_i\left(b+w_1x_i\right) \cancel{-y_i \log\left(1+e^{b+w_1x_i}\right)} - \log\left(1+e^{b+w_1x_i}\right) \\
&\qquad \cancel{+y_i\left(1+e^{b+w_1x_i}\right)}\Big]
\end{aligned}
\tag{20}
$$

$$
\mathrm{NLL}(\boldsymbol{w}) = -\sum_i \left[y_i\left(b+w_1x_i\right) - \log\left(1+e^{b+w_1x_i}\right)\right]
$$

Maximizing the log-likelihood

To find $\hat{\boldsymbol{w}}$, we find the derivative of NLL($\boldsymbol{w}$), set it to zero and solve the resulting **score equations**:

$$\frac{\partial \text{NLL}}{\partial \boldsymbol{w}} = -\frac{\partial}{\partial \boldsymbol{w}} \sum_i \left[y_i \left(b + w_1 x_i \right) - \log \left(1 + e^{b + w_1 x_i} \right) \right]$$

$$\begin{pmatrix} \frac{\partial \text{NLL}}{\partial b} \\ \frac{\partial \text{NLL}}{\partial w_1} \end{pmatrix} = - \begin{pmatrix} \sum_i \left[y_i - \frac{e^{b+w_1 x_i}}{1+e^{b+w_1 x_i}} \right] \\ \sum_i \left[x_i y_i - \frac{x_i \left(e^{b+w_1 x_i} \right)}{1+e^{b+w_1 x_i}} \right] \end{pmatrix} \tag{21}$$

$$\begin{pmatrix} \frac{\partial \text{NLL}}{\partial b} \\ \frac{\partial \text{NLL}}{\partial w_1} \end{pmatrix} = - \begin{pmatrix} \sum_i \left[y_i - p(x_i) \right] \\ \sum_i \left[x_i \left(y_i - p(x_i) \right) \right] \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \end{pmatrix}$$

This is a system of two *nonlinear* equations in $\boldsymbol{w}$ which can be solved via the **Newton-Raphson** method.

Alternatively, we can use the **gradient descent** approach to directly minimize NLL.

Maximum likelihood estimation (MLE) in logistic regression

Recall the negative log-likelihood function for the binomial logistic regression case:

$$\text{NLL}(\boldsymbol{w}) = -\sum_{i} \left[y_i \left(b + w_1 x_i \right) - \log \left(1 + e^{b + w_1 x_i} \right) \right] \tag{22}$$

The optimal $\hat{\boldsymbol{w}}$ which minimizes $\text{NLL}(\boldsymbol{w})$ is the **maximum likelihood estimate**.

Also recall the derivative of NLL:

$$\nabla_{\boldsymbol{w}} \text{NLL} = \begin{pmatrix} \frac{\partial \text{NLL}}{\partial b} \\ \frac{\partial \text{NLL}}{\partial w_1} \end{pmatrix} = \begin{pmatrix} -\sum_i \left[y_i - p(x_i) \right] \\ -\sum_i \left[x_i \left(y_i - p(x_i) \right) \right] \end{pmatrix} \tag{23}$$

We can use either Newton-Raphson or gradient *descent* to *minimize* NLL.

NLL and entropy

We can show that the NLL is equal to the sum of the **binary cross entropy** of y_i and $p(y = 1|\boldsymbol{x}_i)$ over N:

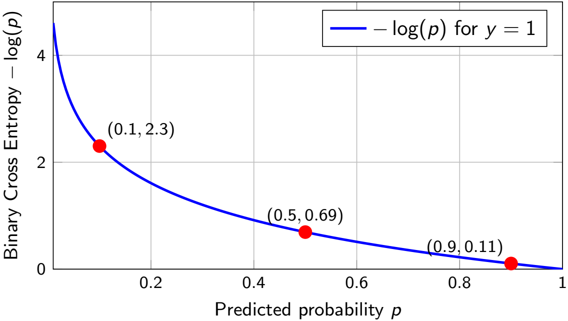
$$\mathbb{H}_i(y_i, p_i) = -[y_i \log p_i + (1 - y_i) \log(1 - p_i)] \tag{24}$$

Note that $p_i = \boldsymbol{\sigma}(\boldsymbol{w}^\top \boldsymbol{x}_i)$.

- Binary cross-entropy quantifies how far your predicted probabilities are from the actual binary labels.

Binary Cross Entropy vs. p (when y=1)

For a true label $y = 1$, the binary cross entropy is $\mathbb{H}(1, p) = -\log(p)$. This function penalizes predictions that are far from the true label:



Key observations:

- As $p \to 1$: cross entropy $\to 0$ (low penalty for correct prediction)
- As $p \to 0$: cross entropy $\to \infty$ (high penalty for incorrect prediction)
- The function is convex, ensuring unique minimum in optimization

Example: predicting spam

a. If the email is spam (y=1) and you predict 90% probability of spam (p=0.9), find the binary cross entropy (BCE):

$$\text{BCE} = -[1 \times \log(0.9) + 0 \times \log(0.1)] = -\log(0.9) \approx 0.105 \quad \text{(low loss - good!)}$$

b. If the email is spam (y=1) but you predict only 10% probability of spam (p=0.1), find the BCE.

$$\text{BCE} = -[1 \times \log(0.1) + 0 \times \log(0.9)] = -\log(0.1) \approx 2.303 \quad \text{(high loss - bad!)}$$

Gradient descent for MLE in logistic regression

This approach only requires the first derivative:

$$\boldsymbol{w}_{k+1} = \boldsymbol{w}_k - \rho \nabla \mathrm{NLL}(\boldsymbol{w}_k) \tag{25}$$

Thus, to find $\hat{\boldsymbol{w}}$ we iterate using:

$$\begin{pmatrix} \boldsymbol{w}_{0,k} \\ \boldsymbol{w}_{1,k} \end{pmatrix} = \begin{pmatrix} \boldsymbol{w}_{0,k} \\ \boldsymbol{w}_{1,k} \end{pmatrix} + \rho \begin{pmatrix} \sum_i \left[y_i - p(x_i) \right] \\ \sum_i \left[x_i \left(y_i - p(x_i) \right) \right] \end{pmatrix} \tag{26}$$

Because the negative log-likelihood is *convex*, and thus a *minimization* problem, we *descend* the function and thus *subtract* the scaled derivative.

Note

- The gradient descent method does not require a second derivative
- However, it may require more iterations to converge than Newton-Raphson

Newton-Raphson approach for MLE in logistic regression

The optimal point $\hat{\boldsymbol{w}}$ is given by the root of the equation $\nabla_{\boldsymbol{w}}\mathrm{NLL} = 0$.

Applying Newton-Raphson, the update step is:

$$\boldsymbol{w}_{k+1} = \boldsymbol{w}_k - \boldsymbol{H}^{-1}_{\boldsymbol{w}_k}(\mathrm{NLL})\nabla_{\boldsymbol{w}_k}\mathrm{NLL}(\boldsymbol{w}_k) \tag{27}$$

The operator $\boldsymbol{H}^{-1}$ represents the inverse **Hessian** (second derivative) matrix of NLL with respect to $\boldsymbol{w}$:

$$\boldsymbol{H}_{\boldsymbol{w}_k}(\mathrm{NLL}) = \nabla^2_{\boldsymbol{w}_k}\mathrm{NLL} = \begin{pmatrix} \frac{\partial^2 \mathrm{NLL}(\boldsymbol{w})}{\partial b^2} & \frac{\partial^2 \mathrm{NLL}(\boldsymbol{w})}{\partial b w_1} \\ \frac{\partial^2 \mathrm{NLL}(\boldsymbol{w})}{\partial w_1 b} & \frac{\partial^2 \mathrm{NLL}(\boldsymbol{w})}{\partial {w_1}^2} \end{pmatrix} \tag{28}$$

Note that (27) is just the matrix representation of the 1-D case:

$$\boldsymbol{w}_{k+1} = \boldsymbol{w}_k - \frac{\mathrm{NLL}'(\boldsymbol{w}_k)}{\mathrm{NLL}''(\boldsymbol{w}_k)} \tag{29}$$

Newton-Raphson approach for MLE (cont.)

We can work out each component of the second derivative:

$$\frac{\partial^2 \mathrm{NLL}(\boldsymbol{w})}{\partial b^2} = \sum_i p(x_i)(1 - p(x_i)) \tag{30}$$

$$\frac{\partial^2 \mathrm{NLL}(\boldsymbol{w})}{\partial b w_1} = \sum_i x_i p(x_i)(1 - p(x_i)) \tag{31}$$

$$\frac{\partial^2 \mathrm{NLL}(\boldsymbol{w})}{\partial {w_1}^2} = \sum_i x_i^2 p(x_i)(1 - p(x_i)) \tag{32}$$

The complete update can then be shown as:

$$\begin{pmatrix} \boldsymbol{w}_{0,k+1} \\ \boldsymbol{w}_{1,k+1} \end{pmatrix} = \begin{pmatrix} \boldsymbol{w}_{0,k} \\ \boldsymbol{w}_{1,k} \end{pmatrix} - \left[\begin{pmatrix} \frac{\partial^2 \mathrm{NLL}(\boldsymbol{w})}{\partial b^2} & \frac{\partial^2 \mathrm{NLL}(\boldsymbol{w})}{\partial b \partial w_1} \\ \frac{\partial^2 \mathrm{NLL}(\boldsymbol{w})}{\partial w_1 \partial b} & \frac{\partial^2 \mathrm{NLL}(\boldsymbol{w})}{\partial {w_1}^2} \end{pmatrix}^{-1} \begin{pmatrix} \frac{\partial \mathrm{NLL}}{\partial b} \\ \frac{\partial \mathrm{NLL}}{\partial w_1} \end{pmatrix} \right]_{\boldsymbol{w}_k} \tag{33}$$

Alternatively:

$$\boldsymbol{w}_{k+1} = \boldsymbol{w}_k - \left[\left(\frac{\partial^2 \mathrm{NLL}(\boldsymbol{w})}{\partial \boldsymbol{w} \partial \boldsymbol{w}^\top} \right)^{-1} \frac{\partial \mathrm{NLL}(\boldsymbol{w})}{\partial \boldsymbol{w}} \right]_{\boldsymbol{w}_k} \tag{34}$$

Compact matrix representation of derivatives

Recall in 1D: $\boldsymbol{w} = \begin{pmatrix} b \\ w_1 \end{pmatrix}$. If we denote: $\boldsymbol{x}_i = \begin{pmatrix} 1 \\ x_i \end{pmatrix}$ Then we can write:

$$\begin{aligned} \frac{\partial \mathrm{NLL}(\boldsymbol{w})}{\partial \boldsymbol{w}} &= \begin{pmatrix} -\sum_i \left[y_i - p(x_i)\right] \\ -\sum_i \left[x_i \left(y_i - p(x_i)\right)\right] \end{pmatrix} = -\begin{pmatrix} 1 & x_1 \\ 1 & x_2 \\ \vdots & \vdots \\ 1 & x_n \end{pmatrix}^T \begin{pmatrix} y_1 - p(x_1) \\ y_2 - p(x_2) \\ \vdots \\ y_n - p(x_n) \end{pmatrix} \\ &= -\boldsymbol{X}^T(\boldsymbol{y} - \boldsymbol{p}) \end{aligned} \tag{35}$$

Compact matrix representation of derivatives (cont.)

We can also decompose the Hessian as:

$$\frac{\partial^2 \text{NLL}(\boldsymbol{w})}{\partial \boldsymbol{w} \partial \boldsymbol{w}^\top} = \begin{pmatrix} \sum_i p(x_i)(1-p(x_i)) & \sum_i x_i p(x_i)(1-p(x_i)) \\ \sum_i x_i p(x_i)(1-p(x_i)) & \sum_i x_i^2 p(x_i)(1-p(x_i)) \end{pmatrix} \tag{36}$$
$$= \boldsymbol{X}^T \boldsymbol{S} \boldsymbol{X}$$

where $\boldsymbol{S}$ is a diagonal $N \times N$ matrix:

$$\boldsymbol{S} = \begin{pmatrix} p(x_1)(1-p(x_1)) & 0 & \dots & 0 \\ 0 & p(x_2)(1-p(x_2)) & \dots & 0 \\ \vdots & \vdots & \ddots & 0 \\ 0 & 0 & 0 & p(x_n)(1-p(x_n)) \end{pmatrix} \tag{37}$$

and $\boldsymbol{X}$ is defined as before:

$$\boldsymbol{X}^T = \begin{pmatrix} 1 & 1 & \cdots & 1 \\ x_1 & x_2 & \cdots & x_n \end{pmatrix} \tag{38}$$

Compact matrix representation (cont.)

Putting the previous results together, we can express the update step as:

$$
\begin{aligned}
\boldsymbol{w}_{k+1} &= \boldsymbol{w}_k - \boldsymbol{H}^{-1}\boldsymbol{g}_k \\
&= \boldsymbol{w}_k + (\boldsymbol{X}^T\boldsymbol{W}\boldsymbol{X})^{-1}\boldsymbol{X}^\top(\boldsymbol{y}-\boldsymbol{p}) \\
&= (\boldsymbol{X}^T\boldsymbol{W}\boldsymbol{X})^{-1}(\boldsymbol{X}^T\boldsymbol{W}\boldsymbol{X})\boldsymbol{w}_k + (\boldsymbol{X}^T\boldsymbol{W}\boldsymbol{X})^{-1}\boldsymbol{X}^T(\boldsymbol{y}-\boldsymbol{p}) \\
&= (\boldsymbol{X}^T\boldsymbol{W}\boldsymbol{X})^{-1}\boldsymbol{X}^T\boldsymbol{W}\left(\boldsymbol{X}\boldsymbol{w}_k + \boldsymbol{W}^{-1}(\boldsymbol{y}-\boldsymbol{p})\right) \\
&= (\boldsymbol{X}^T\boldsymbol{W}\boldsymbol{X})^{-1}\boldsymbol{X}^T\boldsymbol{W}\boldsymbol{z}
\end{aligned}
\tag{39}
$$

where the adjusted response $\boldsymbol{z}$ is given as:

$$
\boldsymbol{z} = \boldsymbol{X}\boldsymbol{w}_k + \boldsymbol{W}^{-1}(\boldsymbol{y}-\boldsymbol{p}) \tag{40}
$$

In this form, the estimation is also called iteratively reweighted least squares (IRLS).

OLS, WLS and IRLS

- Recall the OLS estimate:

$$\hat{\boldsymbol{w}} = (\boldsymbol{X}^T\boldsymbol{X})^{-1}\boldsymbol{X}^T\boldsymbol{y} \tag{41}$$

  if $\boldsymbol{y}$ is the response.
- Also recall that the weighted least squares (WLS) is given by:

$$\hat{\boldsymbol{w}} = (\boldsymbol{X}^T\boldsymbol{W}\boldsymbol{X})^{-1}\boldsymbol{X}^T\boldsymbol{W}y \tag{42}$$

- In logistic regression, the coefficients can be found via the Newton-Raphson update, which can be specified as:

$$\boldsymbol{w}_{k+1} = (\boldsymbol{X}^T\boldsymbol{W}\boldsymbol{X})^{-1}\boldsymbol{X}^T\boldsymbol{W}\boldsymbol{z} \tag{43}$$

  where $\boldsymbol{z}$ is given as:

$$\boldsymbol{z} = \boldsymbol{X}\boldsymbol{w}_k + \boldsymbol{W}^{-1}(\boldsymbol{y} - \boldsymbol{p}) \tag{44}$$

  - Note that the update step is identical in form to the WLS estimator
  - However, $\boldsymbol{W}$ and $\boldsymbol{z}$ change in each iteration, hence the name iteratively reweighted least squares (IRLS)

Summary

- The binary logistic regression model is given by:

$$p(y = 1|\boldsymbol{x}; \boldsymbol{\theta}) = \frac{1}{1 + e^{-\boldsymbol{w}^\top \boldsymbol{x}}} \tag{45}$$

- The negative log-likelihood of a sample of N observations in the binomial response case is:

$$\text{NLL}(\boldsymbol{w}) = \sum_i \left[y_i \left(b + w_1 x_i \right) - \log \left(1 + e^{b + w_1 x_i} \right) \right] \tag{46}$$

- Based on the principle of maximum likelihood, the estimate $\hat{\boldsymbol{w}}$ is given by the minimizing NLL.
- This can be solved via gradient descent or Newton-Raphson.

Summary (cont.)

Gradient descent update for logistic regression

$$\boldsymbol{w}_{k+1} = \boldsymbol{w}_k - \lambda \nabla \mathrm{NLL}(\boldsymbol{w}_k) \tag{47}$$

Newton-Raphson update for logistic regression

$$\boldsymbol{w}_{k+1} = \boldsymbol{w}_k - \boldsymbol{H}_{\boldsymbol{w}_k}^{-1}(\mathrm{NLL}) \nabla_{\boldsymbol{w}_k} \mathrm{NLL}(\boldsymbol{w}_k) \tag{48}$$

This can be rewritten as:

$$\boldsymbol{w}_{k+1} = (\boldsymbol{X}^T \boldsymbol{W} \boldsymbol{X})^{-1} \boldsymbol{X}^T \boldsymbol{W} \boldsymbol{z} \tag{49}$$

where the adjusted response $\boldsymbol{z}$ is given as:

$$\boldsymbol{z} = \boldsymbol{X} \boldsymbol{w}_k + \boldsymbol{W}^{-1}(\boldsymbol{y} - \boldsymbol{p}) \tag{50}$$

In this form, the estimation is also called iteratively reweighted least squares (IRLS).

Other considerations

- MAP estimation: weight decay/regularization to make NLL convex (have unique solution). We define the **penalized negative log-likelihood** PNLL as:

$$\text{PNLL}(\boldsymbol{w}) = \text{NLL}(\boldsymbol{w}) + \lambda \boldsymbol{w}^\top \boldsymbol{w} \tag{51}$$

  where λ is the decay parameter.
- Thus: $\nabla_{\boldsymbol{w}} \text{PNLL}(\boldsymbol{w}) = \boldsymbol{g}(\boldsymbol{w}) + 2\lambda \boldsymbol{w}$
- And: $\nabla^2_{\boldsymbol{w}} \text{PNLL}(\boldsymbol{w}) = \boldsymbol{H}(\boldsymbol{w}) + 2\lambda \boldsymbol{I}$

Reading assignments

- **PMLCE** 9.2
- **PMLI** 10.1-3
- **ESL** 4.4